Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	771	dependency near graph	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/17 13:38
L2	153	1 and algorithm and pointer and node	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/17 14:44
L3	2	(("5808625") or ("5929864")).PN.	US-PGPUB; USPAT	OR	OFF	2005/05/17 14:44
L4	7	("5808625").URPN.	USPAT	OR	ON	2005/05/17 15:21
L5	2	(("5684807") or ("5972314")).PN.	US-PGPUB; USPAT	OR	OFF	2005/05/17 15:21
L6	1	5 and parameter	US-PGPUB; USPAT	OR	ON	2005/05/17 15:21
S1	85	pointer same algorithm same node same execut$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/17 13:29
S2	17	("5684807").URPN.	USPAT	OR	ON	2005/05/12 15:56
S3	1	("4972314").PN.	US-PGPUB; USPAT	OR	OFF	2005/05/12 16:43
S4	1	S3 and pointer	US-PGPUB; USPAT	OR	ON	2005/05/12 16:06
S5	1	S3 and self	US-PGPUB; USPAT	OR	ON	2005/05/12 16:26
S6	1	S3 and execut$4	US-PGPUB; USPAT	OR	ON	2005/05/12 16:26
S7	2	(("4972314") or ("5684807")).PN.	US-PGPUB; USPAT	OR	OFF	2005/05/12 17:44
S8	0	S7 and default	US-PGPUB; USPAT	OR	ON	2005/05/12 17:44
S9	1	S7 and input	US-PGPUB; USPAT	OR	ON	2005/05/12 17:44
S10	1	S7 and input and parameter	US-PGPUB; USPAT	OR	ON	2005/05/12 17:50

S11	770	dependency near graph	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 17:51
S12	367	S11 and algorithm and execut$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 17:51
S13	56	S12 and pointer and self and evaluat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 17:52

Terms used **graph** and **algorithm** and **execute**

Found **61,046** of **154,226**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Transitive closure algorithms based on graph traversal

Yannis Ioannidis, Raghu Ramakrishnan, Linda Winger

September 1993 **ACM Transactions on Database Systems (TODS)**, Volume 18 Issue 3

Full text available: pdf(4.34 MB) Additional Information: full citation, abstract, references, citings, index terms

Several graph-based algorithms have been proposed in the literature to compute the transitive closure of a directed graph. We develop two new algorithms (Basic_TC and Gobal_DFTC) and compare the performance of their implementations in a disk-based environment with a well-known graph-based algorithm proposed by Schmitz. Our algorithms use depth-first search to traverse a graph and a technique called marking to avoid processing some of the arcs in the graph. They compute the ...

Keywords: depth-first search, node reachability, path computations, transitive closure

2 Distributed visualization of graph algorithms

Alexander A. Sherstov

January 2003 **ACM SIGCSE Bulletin , Proceedings of the 34th SIGCSE technical symposium on Computer science education**, Volume 35 Issue 1

Full text available: pdf(218.85 KB) Additional Information: full citation, abstract, references, index terms

DisViz is a visualization tool designed to assist students in learning graph algorithms, an important topic in the undergraduate curriculum. DisViz is intended for collaborative use by a group of students over a classroom network. This visualization system views network hosts as graph nodes and the socket connections among them, as graph edges. In typical usage, every student runs a copy of DisViz on his/her local machine. These applications detect each other's presence on the network and coordi ...

Keywords: computer science education, distributed computing, graph algorithms, simulation and modeling, visualization

3 Gprof: A call graph execution profiler

Susan L. Graham, Peter B. Kessler, Marshall K. Mckusick

June 1982 **ACM SIGPLAN Notices , Proceedings of the 1982 SIGPLAN symposium on Compiler construction**, Volume 17 Issue 6

Full text available: pdf(684.69 KB) Additional Information: full citation, abstract, references, citings, index terms

Large complex programs are composed of many small routines that implement abstractions for the routines that call them. To be useful, an execution profiler must attribute execution time in a way that is significant for the logical structure of a program as well as for its textual decomposition. This data must then be displayed to the user in a convenient and informative way. The gprof profiler accounts for the running time of called routines in the running time of the routines ...

4 A framework for call graph construction algorithms

David Grove, Craig Chambers

November 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 23 Issue 6

Full text available: pdf(1.36 MB) Additional Information: full citation, abstract, references, citings, index terms

A large number of call graph construction algorithms for object-oriented and functional languages have been proposed, each embodying different tradeoffs between analysis cost and call graph precision. In this article we present a unifying framework for understanding call graph construction algorithms and an empirical comparison of a representative set of algorithms. We first present a general parameterized algorithm that encompasses many well-known and novel call graph construction algorithms. W ...

Keywords: Call graph construction, control flow analysis, interprocedural analysis

5 A Fast Implementation of the Minimum Degree Algorithm Using Quotient Graphs

Alan George, Joseph W. H. Liu

September 1980 **ACM Transactions on Mathematical Software (TOMS)**, Volume 6 Issue 3

Full text available: pdf(1.20 MB) Additional Information: full citation, references, citings, index terms

6 Visualization of algorithms: Algorithm animation using shape analysis: visualising abstract executions

Dierk Johannes, Raimund Seidel, Reinhard Wilhelm

May 2005 **Proceedings of the 2005 ACM symposium on Software visualization**

Full text available: pdf(206.49 KB) Additional Information: full citation, abstract, references, index terms

This paper describes progress in a non-traditional approach to algorithm animation. We visualise the abstract execution of an algorithm instead of animating the algorithm for executions on concrete input data. Algorithms under consideration are imperative pointer-based algorithms. For our purpose, we need to compute invariants which describe the state of the heap. The most powerful method to accomplish this task automatically is shape analysis. However, the output of a shape analysis can get con ...

Keywords: abstract execution, algorithm animation, invariant, shape analysis, tree, visualisation

7 An Efficient Implementation of Edmonds' Algorithm for Maximum Matching on Graphs

Harold N. Gabow

April 1976 **Journal of the ACM (JACM)**, Volume 23 Issue 2

Full text available: pdf(847.49 KB) Additional Information: full citation, abstract, references, citings, index terms

A matching on a graph is a set of edges, no two of which share a vertex. A maximum matching contains the greatest number of edges possible. This paper presents an efficient implementation of Edmonds' algorithm for finding a maximum matching. The computation time is proportional to V3, where V is the number of vertices; previous implementations of Edmonds' algorithm have computation time proportional to V4

8 A heuristic multiprocessor scheduling algorithm for creating near-optimal schedules using task system graphs

Farideh A. Samadzadeh, G. E. Hedrick

March 1992 **Proceedings of the 1992 ACM/SIGAPP symposium on Applied computing: technological challenges of the 1990's**

Full text available: pdf(686.75 KB) Additional Information: full citation, references, index terms

9 Faster scaling algorithms for general graph matching problems

Harold N. Gablow, Robert E. Tarjan
October 1991 **Journal of the ACM (JACM)**, Volume 38 Issue 4

Full text available: pdf(2.87 MB) Additional Information: full citation, references, citings, index terms, review

Keywords: augmenting path, blossoms matching, network optimisation, scaling

10 Detecting concurrently executed pairs of statements using an adapted MHP algorithm
Zhenqiang Chen, Baowen Xu, Huiming Yu
September 2001 **ACM SIGAda Ada Letters , Proceedings of the 2001 annual ACM SIGAda international conference on Ada**, Volume XXI Issue 4

Full text available: pdf(48.64 KB) Additional Information: full citation, abstract, references, citings, index terms

Information about which pairs of statements in a program can be executed concurrently is important for improving the accuracy of dataflow analysis, optimizing programs, and detecting errors. This paper presents a new method named Concurrent Control Flow Graph for representing concurrent Ada 95 programs in a simple and precise way. Based on this method, we develop an Adapted MHP algorithm that can statically detect all pairs of statements that may be executed concurrently. This algorithm checks n ...

Keywords: Concurrent program, MHP algorithm, control flow graph, program analysis

11 A graph partitioning algorithm by node separators
Joseph W. H. Liu
September 1989 **ACM Transactions on Mathematical Software (TOMS)**, Volume 15 Issue 3

Full text available: pdf(1.64 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A heuristic graph partitioning scheme is presented to determine effective node separators for undirected graphs. An initial separator is first obtained from the minimum degree ordering, an algorithm designed originally to produce fill-reducing orderings for sparse matrices. The separator is then improved by an iterative strategy based on some known results from bipartite graph matching. This gives an overall practical scheme in partitioning graphs. Experimental results are provided to demon ...

12 An Ada deadlock testing algorithm based on control and communication flow graphs
Anneliese von Mayrhauser, Shao-Ming Simon Hsueh
December 1992 **Proceedings of the conference on TRI-Ada '92**

Full text available: pdf(629.21 KB) Additional Information: full citation, references, citings, index terms

13 Design and evaluation of compiler algorithms for pre-execution
Dongkeun Kim, Donald Yeung
October 2002 **Proceedings of the 10th international conference on Architectural support for programming languages and operating systems**, Volume 37 , 36 , 30 Issue 10 , 5 , 5

Full text available: pdf(1.43 MB) Additional Information: full citation, abstract, references, citings

Pre-execution is a promising latency tolerance technique that uses one or more helper threads running in spare hardware contexts ahead of the main computation to trigger long-latency memory operations early, hence absorbing their latency on behalf of the main computation. This paper investigates a source-to-source C compiler for extracting pre-execution thread code automatically, thus relieving the programmer or hardware from this onerous task. At the heart of our compiler are three algorithms. ...

14 A study of source-level compiler algorithms for automatic construction of pre-execution code
Dongkeun Kim, Donald Yeung
August 2004 **ACM Transactions on Computer Systems (TOCS)**, Volume 22 Issue 3

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, index terms

Pre-execution is a promising latency tolerance technique that uses one or more helper threads running in spare hardware contexts ahead of the main computation to trigger long-latency memory operations early, hence absorbing their latency on behalf of the main computation. This article investigates several source-to-source C compilers for extracting pre-execution thread code automatically, thus relieving the programmer or hardware from this onerous task. We present an aggressive profile-driven co ...

Keywords: Data prefetching, memory-level parallelism, multithreading, pre-execution, prefetch conversion, program slicing, speculative loop parallelization

15 A graph-theoretic model for optimizing queries involving methods

Chiang Lee, Chi-Sheng Shih, Yaw-Huei Chen

April 2001 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 9 Issue 4

Full text available: pdf(266.86 KB) Additional Information: full citation, abstract, index terms

Traditional algorithms for optimizing the execution order of joins are no more valid when selections and projections involve methods and become very expensive operations. Selections and projections could be even more costly than joins such that they are pulled above joins, rather than pushed down in a query tree. In this paper, we take a fundamental look at how to approach query optimization from a top-down design perspective, rather than trying to force one model to fit into another. We present ...

Keywords: Graph model, Method query, Object-oriented databases, Query optimization, Spanning tree

16 SGTNE: semi-global time of the next event algorithm

Azzedine Boukerche, Carl Tropper

July 1995 **ACM SIGSIM Simulation Digest , Proceedings of the ninth workshop on Parallel and distributed simulation**, Volume 25 Issue 1

Full text available: pdf(1.30 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

This paper describes an extension of the TNE algorithm, the objective of which is to increase its parallelism and to break the inter-processor deadlocks inherent with the use of TNE. The algorithm, which we call the SGTNE algorithm (Semi Global TNE), is executed over a cluster of processors as opposed to TNE, which is executed over a cluster of processes assigned to a single processor. SGTNE helps to break the inter-processor deadlocks by executing a shortest path algorithm over a snapshot ...

Keywords: FCFS queueing network model, SGTNE, discrete event simulation, inter-processor deadlock, parallel algorithms, parallelism, partitioning, performance results, shortest path algorithm

17 An efficient search algorithm to find the elementary circuits of a graph

James C. Tiernan

December 1970 **Communications of the ACM**, Volume 13 Issue 12

Full text available: pdf(486.65 KB) Additional Information: full citation, references, citings

Keywords: algorithm, circuit search algorithm, graph theory, path search algorithm, searching

18 An efficient register optimization algorithm for high-level synthesis from hierarchical behavioral specifications

Ranga Vemuri, Srinivas Katkoori, Meenakshi Kaul, Jay Roy
January 2002 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 7 Issue 1
Full text available: pdf(571.24 KB) Additional Information: full citation, abstract, references, index terms

We address the problem of register optimization that arises during high-level synthesis from modular hierarchical behavioral specifications. Register optimization is the process of grouping carriers such that each group can be safely allocated to a hardware register. Global register optimization by inline expansion involves flattening the module hierarchy and using a heuristic register optimization procedure on the flattened description. Although inline expansion yields a near-optimal number of ...

Keywords: Behavioral synthesis, hardware description languages, hierarchical specifications, high-level synthesis, lifecycle analysis, register optimization

19 Algorithm 447: efficient algorithms for graph manipulation
John Hopcroft, Robert Tarjan
June 1973 **Communications of the ACM**, Volume 16 Issue 6
Full text available: pdf(739.68 KB) Additional Information: full citation, abstract, references, citings

Efficient algorithms are presented for partitioning a graph into connected components, biconnected components and simple paths. The algorithm for partitioning of a graph into simple paths of iterative and each iteration produces a new path between two vertices already on paths. (The start vertex can be specified dynamically.) If V is the number of vertices and E is the number of edges, each algorithm requires time and space proportional to max (V, E ...

Keywords: analysis of algorithms, graph manipulation, graphs

20 A modular technique for the design of efficient distributed leader finding algorithms
E. Korach, S. Kutten, S. Moran
January 1990 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 12 Issue 1
Full text available: pdf(1.52 MB) Additional Information: full citation, abstract, references, citings, index terms

A general, modular technique for designing efficient leader finding algorithms in distributed, asynchronous networks is developed. This technique reduces the problem of efficient leader finding to a simpler problem of efficient serial traversing of the corresponding network. The message complexity of the resulting leader finding algorithms is bounded by [f(n) + n) (log2k + 1) (or (f(

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player